An End-to-End Sparse Coding

Joey Tianyi Zhou [1] **Ivor W. Tsang** [2] **Sinno Jialin Pan** [3] **Zheng Qin** [1] **Rick Siow Mong Goh** [1]

Abstract

Learned ISTA (LISTA) is a recently proposed method to achieve sparse codes by unfolding the iterative hard-thresholding algorithm (ISTA) into a simple recurrent neural network (RNN). LISTA's performance, however, depends on the efficiency of precomputed sparse code, as they are prerequisite for loss computation in LISTA. To overcome this limitation, we propose to unfold ISTA into a newly designed Sparse LSTM (SLSTM) network. And it bridges LSTM and sparse coding. Different from traditional sparse coding approaches, SLSTM recasts the optimization procedure as a neural network, thus enables efficient inference both on seen/unseen data. Unlike LISTA, a simple RNN, SLSTM is a LSTM network, which is not an approximation to existing sparse coding methods. SLSTM also differs from traditional LSTMs in the unit structure. SLSTM is specifically designed to sparse coding, whereas LSTMS cannot give sparse representation.

1. Introduction

Sparse coding (SC) has demonstrated its superior decoding ability on uncovering semantic information from noisy and high dimensional data (Wright et al., 2010). It has become a powerful tool for data analytics, especially for image analysis, such as face recognition (Wright et al., 2009), image super-resolution (Zhong et al., 2012), background modeling (Cevher et al., 2008), image classification (Mairal et al., 2008), etc. As the derived optimization problem of sparse coding is non-convex when both dictionary and sparse coefficients are unknown, a classic approach to solving sparse coding is to perform optimization on dictionary learning and sparse approximation alternatively. The two alternating optimization problems can be reduced to the least squared problem under ℓ_2- and ℓ_1- regularization, respectively. Though each individual problem can be efficiently solved by a number of existing optimization tools, the alternating optimization procedure may suffer from several limitations.

Firstly, the procedure optimizes one variable by fixing the other one, which may introduce large fluctuation during parameter updates. This is like a ball oscillates across the slopes of the ravine while only making hesitant progress along the bottom towards a local optimum. Secondly, inference of sparse coding is usually expensive since it requires to solve the well-known LASSO problem with a learned dictionary, which prohibits real-time applications. For example, given an input image, one has to compute the sparse representation for each patch from the image. To speed up inference, Gregor *et al.* (Gregor & LeCun, 2010) recently proposed the Learned ISTA (LISTA) algorithm to unfold the iterative hard-thresholding (ISTA) algorithm (Blumensath & Davies, 2008) into a recurrent neural network (RNN). Though LISTA has shown promisingly empirical results on improving convergence rate, it requires to pre-compute optimal sparse codes using other SC methods in advance. Therefore, the quality of the final sparse codes learned by LISTA for classification/clustering problems highly depends on the pre-computed sparse codes using other methods. In addition, both ISTA and LISTA generate the output of sparse codes exclusively on the previous outputs (at previous iterations or layers). This kind of architecture leads to "microphone phenomena", where the error in the previous layers is propagated and further amplified in the upcoming layers.

To address above limitations, we propose to unfold ISTA into a newly designed Sparse LSTM (SLSTM) network. Different from traditional LSTMs, we introduce an adaptive momentum vector, which act as a gate, into each unit of LSTM to induce sparsity. In this way, LSTM and sparse coding are bridged. Compared with LISTA, which also cast the optimization procedure sparse coding as a forward-backward optimization procedure in neural networks, SLSTM is able to capture long-term "memory" (i.e., useful information for learning parameters from long periods of time), which helps further speed up the convergence rate for learning sparse codes and improve the quality of the learned codes for post-processing prediction problems. The overall op-

[*]Equal contribution [1]Institute of High Performance Computing, A*STAR, Singapore [2]University of Technology Sydney, Austrilia [3]Nanyang Technology University, Singapore. Correspondence to: Joey Tianyi Zhou <zhouty@ihpc.a-star.edu.sg>.

Presented at the ICML 2017 Workshop on Principled Approaches to Deep Learning, Sydney, Australia, 2017.

timization framework for sparse coding using SLSTM is denoted by SC2Net in the sequel, which stands for encoding SC into SLSTM networks. Different from existing methods (e.g. LISTA), our proposed SC2Net does not require pre-computed sparse codes as input, and thus is an end-to-end optimization solution to sparse coding. Moreover, SC2NET is very flexible to be extend to variants of sparse coding with additional task-related regularization loss.

In summary, the main contributions of this paper are summarized as follows,

1. We propose a new LSTM network, named SLSTM, which can naturally connect sparse coding with LSTM. In this way, the learning of sparse coding can be performed through standard forward-background gradient-typed updates. To the best of our knowledge, this is the first work to bridge LSTM network and sparse coding.

2. Different from the previous RNN based sparse coding method, LISTA, the proposed framework SC2NET is not an approximation to existing sparse coding approaches. Instead, SC2NET is a data adaptive approach, and does not require any pre-computed sparse codes, which is truly an end-to-end solution.

2. Preliminaries

Given a data matrix $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_n] \in \mathbb{R}^{d\times n}$, the goal of sparse coding is to learn a shared dictionary $\mathbf{B} = [\mathbf{b}_1, \mathbf{b}_2, \cdots, \mathbf{b}_d] \in \mathbb{R}^{k\times d}$ that can be used to generate sparse codes $\mathbf{S} = [\mathbf{s}_1, \mathbf{s}_2, \cdots, \mathbf{s}_n] \in \mathbb{R}^{k\times n}$ for the input data $\mathbf{X}$. This problem can be solved as follows,

$$\min_{\mathbf{S},\mathbf{B}} \|\mathbf{X} - \mathbf{B}\mathbf{S}\|_F^2 + \lambda\|\mathbf{S}\|_1^2, \;\; s.t. \|\mathbf{b}_i\|^2 \leq 1, \;\; i = 1, \cdots k \tag{1}$$

The alternating optimization is general the most popular method to achieve sparse code. Specifically, the problem 1 is solved by alternating sparse between the sparse approximation and dictionary learning that corresponds to ℓ_1- and ℓ_2- regularized least square problem, respectively.

For a given an input signals $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_n]$, dictionary learning aims to learn a collection of bases $\mathbf{B}$ with fixed sparse codes $\mathbf{S} = [\mathbf{s}_1, \mathbf{s}_2, \cdots, \mathbf{s}_n]$, which generally reduces to the following ℓ_2-constrained optimization problem,

$$\min_{\mathbf{B}} \|\mathbf{X} - \mathbf{B}\mathbf{S}\|_F^2, \;\; s.t. \|\mathbf{b}_i\|^2 \leq 1, \;\; i = 1, \cdots k. \tag{2}$$

This problem is also called as the ridge regression, and exists a closed-form solution. Compared with dictionary learning, sparse approximation attracts much more research attention with goal is to approximate an input signal, $\mathbf{x}$, in terms of a "sparse" combination of fixed bases $\mathbf{B}$. The sparse signal can be recovered by solving,

$$\min_{\mathbf{s}} \|\mathbf{X} - \mathbf{B}\mathbf{S}\|_F^2 + \lambda\|\mathbf{S}\|_1^2. \tag{3}$$

The above problem can be solved by the iterative hard-thresholding (ISTA) algorithm proposed in (Blumensath & Davies, 2008) with theoretical guarantee. To be specific, ISTA decomposes the objective of Problem 3 into two parts: the differentiable part $g(\mathbf{S}) = \|\mathbf{X} - \mathbf{B}\mathbf{S}\|_F^2$ is updated by gradient descent and ℓ_1 part is updated by hard thresholding. The updating formula can be mathematically express as follows,

$$\mathbf{S}^{(t)} = sh_{(\lambda\tau)}(\mathbf{S}^{(t-1)} - \tau\nabla g(\mathbf{S}^{(t-1)})), \tag{4}$$

where the shrinkage function is defined as $sh_{(\lambda\tau)}(\mathbf{S}) = \text{sign}(\mathbf{S})(|\mathbf{S}| - \lambda\tau)_+$. The solution of Problem 4 can then be solved through the following updating rule,

$$\begin{aligned} \mathbf{S}^{(t)} &= sh_{(\lambda\tau)}(\mathbf{S}^{(t-1)} - \tau(\mathbf{B}^\top(\mathbf{B}\mathbf{S}^{(t-1)} - \mathbf{X}))) \quad (5)\\ &= sh_{(\lambda\tau)}(\mathbf{W}_e\mathbf{S}^{(t-1)} + \mathbf{W}_d\mathbf{X}), \quad (6) \end{aligned}$$

where $\mathbf{W}_e = \mathbf{I} - \tau\mathbf{B}^\top\mathbf{B}$, $\mathbf{W}_d = \tau\mathbf{B}^\top$. Gregor *et al.* proposed a neural network based method, termed LISTA (Gregor & LeCun, 2010) to unfold the above ISTA into a simple Recurrent Neural Network (RNN) wherein $\mathbf{W}_e$ and $\mathbf{W}_d$ are decoupled. In other words, $\mathbf{W}_e$ and $\mathbf{W}_d$ are treated as two independent variables during training in LISTA. Such a decoupling makes LISTA different from ISTA and our experiments will show that the decoupling is helpful to enjoy faster convergence speed as well as better optimum. We found that in LISTA the parameters $\mathbf{W}_e, \mathbf{W}_d$ and τ in the non-linear activation function $sh_{(\lambda\tau)}$ are shared in all the layers and updated simultaneously through backpropagation to minimize $\|\mathbf{s} - \mathbf{s}^*\|$, where $\mathbf{s}^*$ is the optimal sparse code precomputed from other existing SC methods like ISTA.

3. SC2Net

The architecture proposed in LISTA has two disadvantages. First, it requires an additional effort to get $\mathbf{s}^*$ thus leading to a high computational cost. Second, LISTA largely depends on the quality of precomputed $\mathbf{s}^*$. The poorly precomputed sparse code $\mathbf{s}^*$ subsequently affects the optimization of sparse representation $\mathbf{s}$. To avoid above two limitations, we propose an optimization framework for SC, in short for SC2Net, which reformulates SC as a LSTM network instead of a simple RNN. Specifically, the sparsity loss and the reconstruction loss are incorporated into the LSTM network to supervise the optimization process without any prior knowledge. For any data point $\mathbf{x}$, the reconstruction loss is constructed through as follows:

$$\|\mathbf{x} - \frac{1}{\tau}\mathbf{W}_d^\top\mathbf{s}\|_F^2 \tag{7}$$

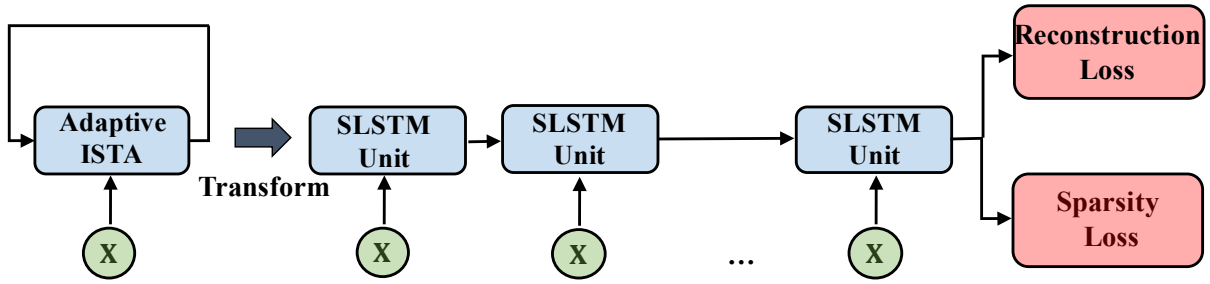

Figure 1. The Architecture of SC2Net.

where $\mathbf{s}$ is the output of encoding part in the network for $\mathbf{x}$ and we apply the relationship $\mathbf{B} = \frac{1}{\tau}\mathbf{W}_d^\top$ from Eqn 6. Here we avoid to learn individual decoding matrix and instead reuse the encoding matrix $\mathbf{W}_d$, which gives two advantages: 1) It maintains the physical meaning of the original formulation; 2) It avoids to learn additional parameters and reduce the computation cost.

The ℓ_1 loss is also considered to control sparsity. The overall cost function for SC2Net is defined as follows, [1],

$$\|\mathbf{x} - \frac{1}{\tau}\mathbf{W}_d^\top \mathbf{s}\|_F^2 + \lambda\|\mathbf{s}\|_1, \tag{8}$$

The architecture of SC2Net is illustrated in Figure 1.

3.1. Adaptive ISTA

ISTA builds the output of sparse codes exclusively on the previous output without considering the historical information. Recently, introduction of the "momentum" into dynamics of stochastic gradient descent (SGD) shows promising improvement on the robustness of SGD thanks to the incorporation of historical updating information (Qian, 1999).

Borrowing the notion of momentum, we introduce the *adaptive momentum vectors* $\mathbf{i}^{(t)}, \mathbf{f}^{(t)}$ into ISTA at time step t, which can be formulated as follows,

$$\tilde{\mathbf{C}}^{(t)} = \mathbf{W} \cdot [\mathbf{S}^{(t-1)}, \mathbf{X}], \tag{9}$$
$$\mathbf{C}^{(t)} = \mathbf{f}^{(t)} \odot \mathbf{C}^{(t-1)} + \mathbf{i}^{(t)} \odot \tilde{\mathbf{C}}^{(t)}, \tag{10}$$
$$\mathbf{S}^{(t)} = sh_{(\lambda\tau)}(\mathbf{C}^{(t)}), \tag{11}$$

where $\mathbf{W} = [\mathbf{W}_e, \mathbf{W}_d]$. Instead of directly feeding the output of ISTA from time step $t-1$ (i.e., $\tilde{\mathbf{C}}^{(t)}$) into ISTA, we also consider the linear combination of $\mathbf{C}^{(t-1)}$ at the previous iteration and $\tilde{\mathbf{C}}^{(t)}$ at the current iteration weighted with adaptive momentum vectors $\mathbf{f}^{(t)}, \mathbf{i}^{(t)}$, respectively. The adaptive momentum vectors allows per-parameter combination of two outputs, which is different from directly applying momentum methods into ISTA. This linear combination perhaps make the output less sparse. Therefore, we feed $\tilde{\mathbf{C}}^{(t)}$ into the shrinkage function again to ensure the outputs to be sparse. We name this method as **adaptive ISTA** in the upcoming sections.

[1] In the experiments, the network is often learned through minimizing the average cost over a set of training samples using a stochastic gradient method.

3.2. Sparse Long Short Term Memory Unit (SLSTM)

Although introducing adaptive momentum vectors brings flexibility in the optimization model, it triggers another big challenge, i.e., how to determine the values of adaptive momentum vectors $\mathbf{f}^{(t)}, \mathbf{i}^{(t)}$. To address this challenge, we propose a novel unit named ***Sparse Long Short Term Memory Unit (SLSTM)*** which parametrizes the adaptive momentum vectors with the output of sparse codes at the previous layer as well as input data such that $\mathbf{f}^{(t)}, \mathbf{i}^{(t)}$ are learned from data without tedious hand-craft tuning. What makes the problem more interesting is that the iterative steps in adaptive ISTA can be reformulated as an LSTM unit wherein "input gate" and "forget gate" corresponds $\mathbf{i}^{(t)}$ and $\mathbf{f}^{(t)}$ respectively. Different from standard LSTM, we discard "output gate" in LSTM and uses different nonlinear activation function for cell states. The proposed SLSTM unit is formulated as follows,

$$\mathbf{i}^{(t)} = \sigma(\mathbf{W}_i \cdot [\mathbf{S}^{(t-1)}, \mathbf{X}]) \tag{12}$$
$$\mathbf{f}^{(t)} = \sigma(\mathbf{W}_f \cdot [\mathbf{S}^{(t-1)}, \mathbf{X}]) \tag{13}$$
$$\mathbf{o}^{(t)} = \sigma(\mathbf{W}_o \cdot [\mathbf{S}^{(t-1)}, \mathbf{X}]) \tag{14}$$
$$\tilde{\mathbf{C}}^{(t)} = \mathbf{W} \cdot [\mathbf{S}^{(t-1)}, \mathbf{X}] \tag{15}$$
$$\mathbf{C}^{(t)} = \mathbf{f}^{(t)} \odot \mathbf{C}^{(t-1)} + \mathbf{i}^{(t)} \odot \tilde{\mathbf{C}}^{(t)} \tag{16}$$
$$\mathbf{S}^{(t)} = h_{(\mathbf{D},\mathbf{u})}(\mathbf{C}^{(t)}) \tag{17}$$

where $\sigma(\mathbf{x}) = \frac{1}{1+e^{-\mathbf{x}}}$, $h_{(\mathbf{D},\mathbf{u})} = \mathbf{D}(tanh(\mathbf{x}+\mathbf{u}) + tanh(\mathbf{x}-\mathbf{u}))$ and $\mathbf{u}$, $\mathbf{D}$ are a trainable vector and diag-

onal matrix, respectively.

Both ISTA and LISTA build the output of sparse codes exclusively on the previous output. This kind of architecture leads to "error propagation phenomenon" where the error in the first few layers will be propagated and further amplified in the upcoming layers. Furthermore, once the useful information is discarded by the previous layers, the upcoming layers will have no chance to utilize the discarded information. Fortunately, this issue can be alleviated with the usage of "cell" state $\mathbf{C}^{(t)}$ in our method. The "cell" plays as another "eye" to supervise the optimization, which brings two major advantages. 1) It captures long-term dependence from the previous outputs. 2) It automatically accumulates important information and forgets useless or redundant information in the dynamics of neural networks.

4. Experiments

In this section, we conduct experiments to verify the effectiveness of the proposed SLSTM in classification.

4.1. Setup and Data

For fair comparisons, we adopt the same optimizer (i.e. Adadelta (Zeiler, 2012)) to train all neural network based apporaches with a GPU of NVIDIA TITAN X. Moreover, we fix the sparsity parameter $\lambda = 0.1$ for all the tested methods[2]. In other words, all the baselines and the proposed method share the same objective function, while they differ in the optimization approach. [3] In experiments, we compare the proposed SLSTM with ISTA (Blumensath & Davies, 2008), FISTA(Blumensath & Davies, 2008), LISTA (Gregor & LeCun, 2010), and LFISTA (Moreau & Bruna, 2017). FISTA is an accelerated version of ISTA that converges faster, both in theory and in practice. It considers the difference of the last two outputs of the shrinkage function. For non-neural network-based methods (i.e. ISTA and LFISTA), they uses the dictionary learning is implemented by scikit-learn (Pedregosa et al., 2011). LISTA and LFISTA solve SC by unfolding the ISTA and FISTA into a simple RNN, respectively. We carry out experiments using MINST (Lecun & Cortes) and CIFAR-10 (Krizhevsky, 2009). MINST contains 60,000 training images and 10,000 testing images sampled from the digits (0-9), where each image has the resolution of 28×28. The CIFAR-10 dataset is another widely used benchmark dataset for various computer vision tasks, which contains 60,000 $32 \times 32 \times 3$ colour images distributed from 10 classes. Following the experimental setting in (Gregor & LeCun, 2010; Montazer et al., 2012), we rescale all images into the range of $[0, 1]$ and build a dictionary with 100 atoms learned from the original data set.

[2]Noted that, we experimentally found that all the tested methods perform stably when λ ranges from 0.01 to 0.1.

[3]All the experiments can be reproduced through the code at Github. We will make it publicly available after acceptance.

4.2. Classification Comparison

In this section, we investigate the performance of our method for classification. Notice that, our method can be easily extended as a cascade model to get discriminative codes by incorporating the label information into the loss function, for fair comparisons, however, we evaluate the performance of all tested methods in an unsupervised way. More specifically, after getting sparse codes with the tested methods, we train a logistics regression classifier with the extracted sparse codes. Table 1 reports the results from which, we observe that all the methods perform better on MNIST than on CIFAR-10. The possible reasons may be that 1) CIFAR-10 images are more complicated than MNIST graysale images. 2) the sparse representation on original colour CIFAR-10 images discards more discriminative information than MNIST graysale images. Furthermore, SLSTM outperforms all the baselines by a large margin on the both two datasets. This may attribute to the advantages of SLSTM, i.e., the long-term dependence, nonlinearity, and data-driven coding optimization procedure.

Datasets	ISTA	FISTA	LISTA	LFISTA	SLSTM
MNIST	85.25	86.65	85.75	85.55	89.81
CIFAR-10	34.05	35.75	35.12	35.65	39.10

Table 1. Overall Comparison in terms of Classification Accuracy

5. Conclusion

We propose a new architecture which reformulates sparse coding (SC) optimization as a LSTM network, named SC2Net. To show the effectiveness of SC2Net, we propose a new sparse code method (SLSTM) by implementing the well-known ISTA using SC2Net. Extensive experimental results show the effectiveness of SLSTM comparing with several famous methods including ISTA, LISTA, FISTA, and LFISTA.

References

Blumensath, Thomas and Davies, Mike E. Iterative thresholding for sparse approximations. *J. Fourier Anal. Appl.*, 14(5):629–654, 2008.

Cevher, Volkan, Sankaranarayanan, Aswin, Duarte, Marco, Reddy, Dikpal, Baraniuk, Richard, and Chellappa, Rama. Compressive sensing for background subtraction. *ECCV*, pp. 155–168, 2008.

Gregor, Karol and LeCun, Yann. Learning fast approximations of sparse coding. In Fürnkranz, Johannes and Joachims, Thorsten (eds.), *ICML*, pp. 399–406. Omnipress, 2010. URL http://www.icml2010.org/papers/449.pdf.

Krizhevsky, Alex. Learning multiple layers of features from tiny images. 2009.

Lecun, Yann and Cortes, Corinna. The MNIST database of handwritten digits. URL http://yann.lecun.com/exdb/mnist/.

Mairal, Julien, Bach, Francis, Ponce, Jean, Sapiro, Guillermo, and Zisserman, Andrew. Discriminative learned dictionaries for local image analysis. In *CVPR*, pp. 1–8. IEEE, 2008.

Montazer, Gholam Ali, Escalera, Sergio, et al. Error correcting output codes for multiclass classification: Application to two image vision problems. In *AISP*, pp. 508–513. IEEE, 2012.

Moreau, Thomas and Bruna, Joan. Understanding neural sparse coding with matrix factorization. *ICLR*, 2017.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., and Duchesnay, E. Scikit-learn: Machine learning in Python. *J. Mach. Learn. Res.*, 12:2825–2830, 2011.

Qian, Ning. On the momentum term in gradient descent learning algorithms. *Neural networks*, 12(1):145–151, 1999.

Wright, John, Yang, Allen Y, Ganesh, Arvind, Sastry, S Shankar, and Ma, Yi. Robust face recognition via sparse representation. *IEEE Trans. Pattern Anal. Mach. Intell.*, 31(2):210–227, 2009.

Wright, John, Ma, Yi, Mairal, Julien, Sapiro, Guillermo, Huang, Thomas S, and Yan, Shuicheng. Sparse representation for computer vision and pattern recognition. *Proc. IEEE*, 98(6):1031–1044, 2010.

Zeiler, Matthew D. Adadelta: an adaptive learning rate method. *arXiv preprint arXiv:1212.5701*, 2012.

Zhong, Wei, Lu, Huchuan, and Yang, Ming-Hsuan. Robust object tracking via sparsity-based collaborative model. In *CVPR*, pp. 1838–1845. IEEE, 2012.